1. General model

Our project intends to distinguish people with makeup from those who not, which is a classification problem. With our dataset, hundreds of photos of people from different regions, we decided to introduce convolutional neural network(CNN), which is a branch of machine learning popular in fields of classification.

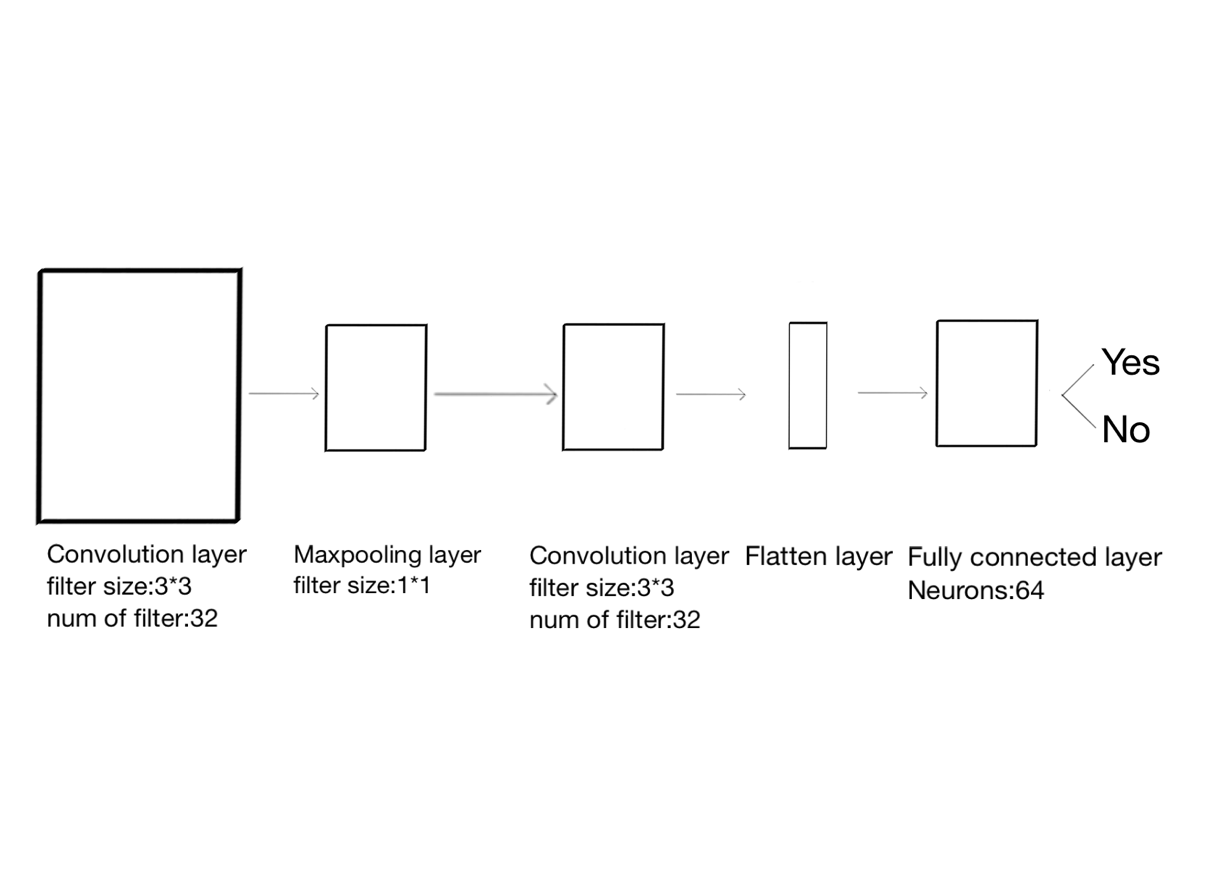
In machine learning, convolutional neural network( CNN) is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery.

CNNs use a variation of multilayer perceptron designed to require minimal preprocessing.[1] They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.[2][3]

Convolutional networks were inspired by biological processes[4] in which the connectivity pattern between neurons is inspired by the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

CNNs use relatively little pre-processing compared to other image classification algorithms, hence, the network learns the filters that in traditional algorithms were hand-engineered. The independence from prior knowledge and human effort in feature design is a major advantage.

Below is the flow chart we use when building CNN model. The input size is m\*n\*Num, indicating we have Num images with size of m\*n, the next layer is a maxpooling layer, and the output size will be (m/2)\*(n/2)\*Num. Then we use a convolution layer with filter size being 3\*3, followed by a flatten layer. Finally, we have a fully connected layer where we can classify images into two categories, which is, with makeup and without makeup.

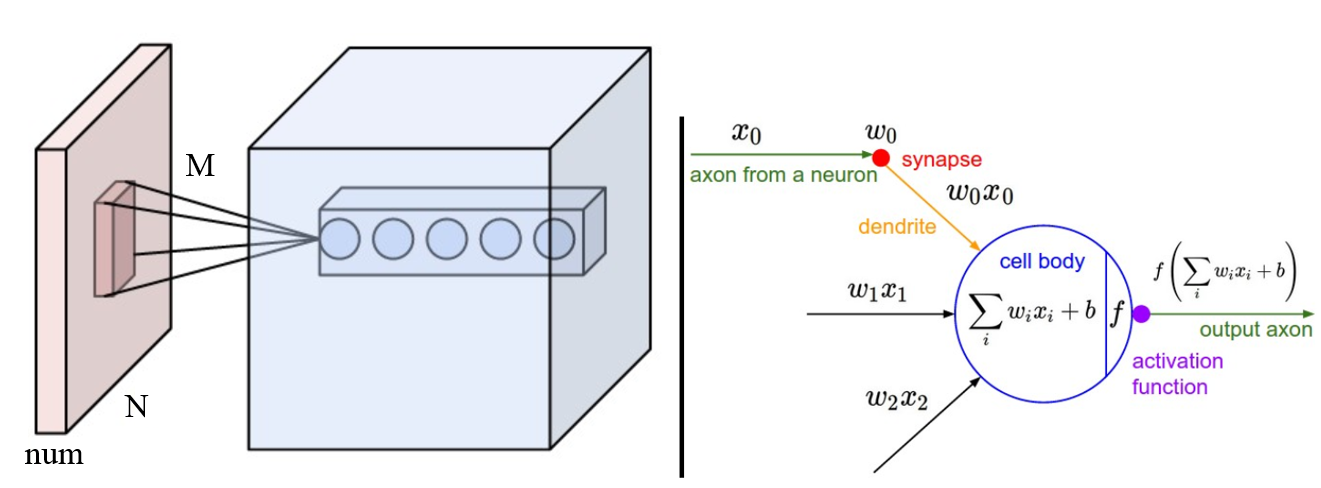


convolutional neural network

1. *LeCun, Yann.*[*"LeNet-5, convolutional neural networks"*](http://yann.lecun.com/exdb/lenet/)*. Retrieved 16 November 2013.*
2. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/Convolutional_neural_network#cite_ref-:0_2-0) [***b***](https://en.wikipedia.org/wiki/Convolutional_neural_network#cite_ref-:0_2-1) *Zhang, Wei (1988).*[*"Shift-invariant pattern recognition neural network and its optical architecture"*](https://drive.google.com/file/d/0B65v6Wo67Tk5Zm03Tm1kaEdIYkE/view?usp=sharing)*. Proceedings of annual conference of the Japan Society of Applied Physics.*
3. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/Convolutional_neural_network#cite_ref-:1_3-0) [***b***](https://en.wikipedia.org/wiki/Convolutional_neural_network#cite_ref-:1_3-1) *Zhang, Wei (1990).*[*"Parallel distributed processing model with local space-invariant interconnections and its optical architecture"*](https://drive.google.com/file/d/0B65v6Wo67Tk5ODRzZmhSR29VeDg/view?usp=sharing)*. Applied Optics.****29****(32): 4790–7.*[*Bibcode*](https://en.wikipedia.org/wiki/Bibcode)*:*[*1990ApOpt..29.4790Z*](http://adsabs.harvard.edu/abs/1990ApOpt..29.4790Z)*.*[*PMID*](https://en.wikipedia.org/wiki/PubMed_Identifier)[*20577468*](https://www.ncbi.nlm.nih.gov/pubmed/20577468)*.*[*doi*](https://en.wikipedia.org/wiki/Digital_object_identifier)*:*[*10.1364/AO.29.004790*](https://doi.org/10.1364%2FAO.29.004790)*.*
4. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/Convolutional_neural_network#cite_ref-robust_face_detection_4-0) [***b***](https://en.wikipedia.org/wiki/Convolutional_neural_network#cite_ref-robust_face_detection_4-1) *Matusugu, Masakazu; Katsuhiko Mori; Yusuke Mitari; Yuji Kaneda (2003).*[*"Subject independent facial expression recognition with robust face detection using a convolutional neural network"*](http://www.iro.umontreal.ca/~pift6080/H09/documents/papers/sparse/matsugo_etal_face_expression_conv_nnet.pdf)*(PDF). Neural Networks.****16****(5): 555–559.*[*doi*](https://en.wikipedia.org/wiki/Digital_object_identifier)*:*[*10.1016/S0893-6080(03)00115-1*](https://doi.org/10.1016%2FS0893-6080%2803%2900115-1)*. Retrieved 17 November 2013.*
5. Model implementation

 To summarize, the Conv Layer:

* Accepts a volume of size 
* Requires four hyperparameters:
  + Number of filters ,
  + their spatial extent ,
  + the stride ,
  + the amount of zero padding .
* Produces a volume of size where:
  + 
  +  (i.e. width and height are computed equally by symmetry)
  + 
* With parameter sharing, it introduces  weights per filter, for a total of  weights and  biases.
* In the output volume, the d-th depth slice (of size ) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of SS, and then offset by d-th bias.

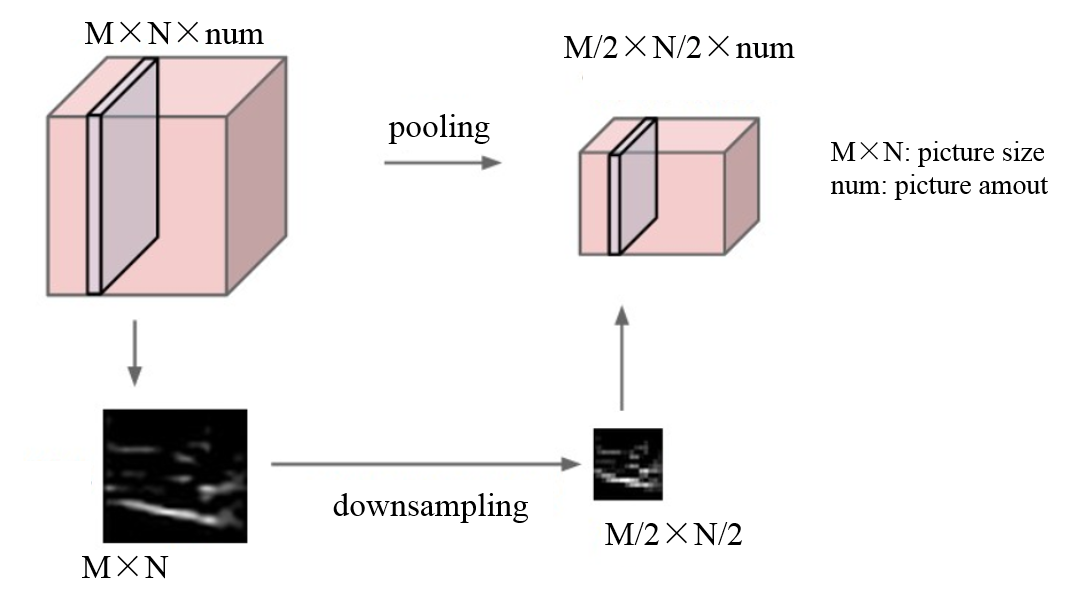


Conv Layer

#### Pooling Layer

#### It is common to periodically insert a Pooling layer in-between successive Conv layers in a ConvNet architecture. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 downsamples every depth slice in the input by 2 along both width and height, discarding 75% of the activations. Every MAX operation would in this case be taking a max over 4 numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged. More generally, the pooling layer:

* Accepts a volume of size 
* Requires two hyperparameters:
  + their spatial extent ,
  + the stride ,
* Produces a volume of size  where:
  + 
  + 
  + 
* Introduces zero parameters since it computes a fixed function of the input
* Note that it is not common to use zero-padding for Pooling layers



Pooling Layer

#### Normalization Layer

Many types of normalization layers have been proposed for use in ConvNet architectures, sometimes with the intentions of implementing inhibition schemes observed in the biological brain. However, these layers have since fallen out of favor because in practice their contribution has been shown to be minimal, if any.

#### Fully-connected layer

Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset.

